

NO-A191 663

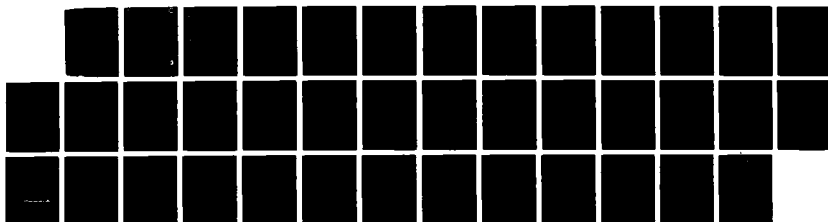
A CONSTRAINT ALGORITHM FOR MAINTAINING RIGID BONDS IN
MOLECULAR DYNAMICS. (U) NAVAL RESEARCH LAB WASHINGTON
DC S G LANBRAKOS ET AL. 04 MAR 88 NRL-NR-6174

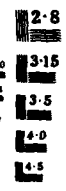
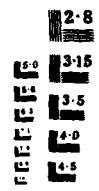
1/1

UNCLASSIFIED

F/G 7/4

NL





UIN FILE LUK

Naval Research Laboratory

Washington, DC 20375-5000

2



NRL Memorandum Report 6174

A Constraint Algorithm for Maintaining Rigid Bonds in Molecular Dynamics Simulations of Large Molecules

S. G. LAMBRAKOS, J. P. BORIS AND E. S. ORAN

*Center for Reactive Flow and Dynamical Systems Branch
Laboratory for Computational Physics and Fluid Dynamics Division*

I. CHANDRASEKHAR* AND M. NAGUMO

*Bio/Molecular Engineering Branch
Chemistry Division*

**Molecular Biophysics Unit
Indian Institute of Science
Bangalore 560-012 India*

March 4, 1988

DTIC
SELECTED
APR 21 1988
S E D

Approved for public release; distribution unlimited.

88 4 20 00 9

AD-A191 663

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NRL Memorandum Report 6174			7a. NAME OF MONITORING ORGANIZATION		
6a. NAME OF PERFORMING ORGANIZATION Naval Research Laboratory		6b. OFFICE SYMBOL (If applicable) Code 4410	7b. ADDRESS (City, State, and ZIP Code)		
6c. ADDRESS (City, State, and ZIP Code) Washington, DC 20375-5000			8a. NAME OF FUNDING/SPONSORING ORGANIZATION Office of Naval Research		
8b. OFFICE SYMBOL (If applicable)			9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code) Arlington, VA 22217			10. SOURCE OF FUNDING NUMBERS		
PROGRAM ELEMENT NO. ONR		PROJECT NO. ONR	TASK NO.	WORK UNIT ACCESSION NO. DN158-074	
11. TITLE (Include Security Classification) A Constraint Algorithm for Maintaining Rigid Bonds in Molecular Dynamics Simulations of Large Molecules					
12. PERSONAL AUTHOR(S) Lambrakos, S.G., Boris, J.P., Oran, E.S., Chandrasekhar, * I. and Negaumo, M.					
13a. TYPE OF REPORT Interim		13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day) 1988 March 4		15. PAGE COUNT 39
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	Constraint algorithm , Polyatomic molecules		
			Molecules dynamics ,		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
<p>We present an efficient new algorithm, the Multiple Constraint Force (MCF) algorithm, for maintaining fixed distances in molecular dynamics simulations of polyatomic molecules. A reversible leapfrog integration scheme is modified to calculate the constraint forces required to maintain fixed interparticle distances (usually corresponding to chemical bonds). The procedure is exact for diatomic systems. It converges in a small number of iterations for polyatomic molecules and scales linearly with the number of bonds to be kept rigid. The MCF algorithm, like the SHAKE algorithm, iteratively corrects the constrained equations of motion. Rather than successively correcting particle positions or inverting a matrix, however, the MCF algorithm iteratively corrects the constraint forces required to maintain the fixed distances. (Kopelman 85)</p>					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Samuel G. Lambrakos			22b. TELEPHONE (Include Area Code) (202) 767-5398		22c. OFFICE SYMBOL Code 4410

CONTENTS

I.	INTRODUCTION	1
II.	DERIVATION OF THE CONSTRAINT FORCE FUNCTION	4
III.	PROCEDURE FOR CALCULATING CONSTRAINT FORCES	9
IV.	TESTS OF THE ALGORITHM	11
V.	CONCLUSION	14
	ACKNOWLEDGEMENTS	17
	REFERENCES	18
	APPENDIX 1 — Derivation of Eq. (2): Maximum Timestep for Accurately Integrating Equations of Motion	29
	APPENDIX 2 — Geometric Interpretation of Eq. (15)	31

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



A CONSTRAINT ALGORITHM FOR MAINTAINING RIGID BONDS IN MOLECULAR DYNAMICS SIMULATIONS OF LARGE MOLECULES

I. INTRODUCTION

Accurate integration of particle trajectories in molecular dynamics (MD) simulations requires timesteps small enough to resolve numerical representations of the second-order differential equation

$$\frac{d^2 X}{dt^2} = F(X, t). \quad (1)$$

Common algorithms for integrating Eq. (1) include Runge-Kutta and other multistep methods, various predictor-corrector methods and central differencing schemes. The Verlet¹, Beeman², and leapfrog algorithms^{3,4,5} are commonly used to integrate the equations of motion in MD simulations. The simplest leapfrog method, which integrates Eq. (1) using only second-order central differences of X and dX/dt , has several advantages. It is reversible with respect to the independent variable (t in Eq. (1)). It requires fewer operations and less computer memory because it does not have to keep track of particle positions at several previous timesteps. Finally, the leapfrog algorithm tends to conserve energy better than other methods.

The maximum timestep size for accurately integrating the equations of motion in MD simulations is given by

$$\delta t = \frac{\alpha}{\max \left[\frac{dF_i}{dX} \right]^{\frac{1}{2}}}, \quad (2)$$

where $\max(dF_i/dX)$ is the maximum of the gradient of the force of all interparticle forces in the system, and α is a parameter related to the accuracy of the numerical integration. A derivation of Eq. (2) is given in Appendix 1. In simulations of molecular systems, the short timescale of intramolecular forces usually determines the timestep. The associated degrees of freedom represent fast modes, such as vibrations. Resolving these modes is generally not important for simulating slower modes such as intermolecular vibrations, torsional angle transitions, or molecular reorientations. When the fastest

degrees of freedom can be ignored, the computational efficiency can be increased by using constraint algorithms that eliminate motions on the fast scales.

Various iterative and noniterative methods have been developed for maintaining rigid bonds in MD simulations of large molecules. For example, two commonly used algorithms are SHAKE⁶⁻¹⁰, which is iterative, and the matrix method,⁶ which is non-iterative. The matrix method inverts a matrix to solve for Lagrangian multipliers that satisfy the constraint conditions, and so becomes computationally expensive for very large molecules. The SHAKE algorithm avoids matrix inversion by iteratively adjusting particle coordinates until the system satisfies all the constraints to within a given tolerance. In addition to maintaining rigid bonds, constraint algorithms must counteract the increasing departure from the fixed distances, called constraint decay, resulting from the accumulation of numerical errors. Iterative algorithms counteract constraint decay implicitly by requiring convergence to within a specified tolerance at each timestep. Deviations in the constrained distances from their initial values are continually checked and corrected. Noniterative algorithms require an explicit scheme for counteracting constraint decay because there is no inherent feedback mechanism for monitoring changes in distance.

Recently Edberg et al.¹¹ developed a noniterative algorithm for maintaining fixed distances between particles. In conjunction with this algorithm, they developed a criterion to ensure that constrained distances only deviate from their assigned values by amounts small enough that the algorithm remains stable. This approach defines penalty functions that monitor constraint deviations. When the penalty functions reach some specified value, the penalties are minimized by correcting the deviations according to Gauss' principle of least constraint¹². By relaxing the constraint slightly, the computational cost is reduced because the accumulation of numerical errors is not corrected every timestep.

In this paper, we present a new algorithm for enforcing holonomic constraints that

can be used in conjunction with the leapfrog algorithm. We show that it is extremely efficient, gives constraint forces explicitly, and provides flexibility for computing constraint forces according to the most efficient programming structures for a given problem or computer. A Multiple Constraint Force (MCF) function is derived using the reversible leapfrog integration algorithm with constraint conditions applied pairwise to all restricted particles. This algorithm is similar to that developed by Memon et al.¹³ in that it is based on the leapfrog integration algorithm and is iterative. Rather than solving for constraint forces in a matrix inversion, however, the MCF algorithm applies the two-body constraint force function iteratively to each restricted distance in the polyatomic molecule. The constraint force is added to the other forces acting on each particle, so the value of the constraint force function decreases with successive iterations. The test simulations using this algorithm show that the MCF algorithm converges rapidly, and numerical errors do not accumulate, so there is no unstable constraint decay. Rather, the constraint error fluctuates stably. The constraint fluctuations are caused by small errors in the constraint forces resulting from: the incomplete convergence of the constraint-force functions when using a small number of iterations; approximations in the constraint force function for the purpose of increasing computational efficiency; and numerical errors due to discretization, roundoff and truncation.

In principle, any two-particle constraint algorithm could serve as the basis of an iterative multiple-constraint algorithm since the evaluation of any two-particle constraint force can include other constraint forces as external forces. For example, Singer et al.¹⁴ have developed an efficient and widely used two-particle constraint algorithm. The Singer algorithm, however, treats the dynamics of the center-of-mass separately from the rotational motion, thus treating the dynamics of the two linked particles implicitly. Consequently, this method does not permit a simple extension to a system with multiple constraints.

II. Derivation of the Constraint Force Function

A constraint force function for maintaining rigid bonds can be derived by applying the leapfrog algorithm to solve Eq. (1) numerically. This is done by modifying the leapfrog procedure to include the forces of constraint, δF_{ij} , so that the position X and the velocity V of each particle are given by:

$$X_i^{n+1} = X_i^n + V_i^{n+\frac{1}{2}} \delta t \quad (3)$$

$$V_i^{n+\frac{1}{2}} = V_i^{n-\frac{1}{2}} + (F_i^n + \delta F_{ij}^n) \frac{\delta t}{m_i}, \quad (4)$$

where δF_{ij}^n is the constraint force for the fixed distance separating particles i and j and F_i^n is the sum of all forces on particle i . The masses of particles i and j are given by m_i and m_j , respectively. The superscripts in Eqs. (3) and (4) indicate that X and V are central differenced in time. After n timesteps of size δt , V and X are computed at times $(n + \frac{1}{2})\delta t$ and $(n + 1)\delta t$, respectively.

The velocity of particle j at step $n + \frac{1}{2}$ is

$$V_j^{n+\frac{1}{2}} = V_j^{n-\frac{1}{2}} + (F_j^n - \delta F_{ij}^n) \frac{\delta t}{m_j}. \quad (5)$$

The constraint force δF_{ij}^n must be such that the condition

$$|X_j^{n+1} - X_i^{n+1}|^2 = l_o^2, \quad (6)$$

is maintained, where $l_o = |l_o|$ is the fixed distance between particles i and j . However, the errors accumulate at each timestep in any real simulation so that

$$|X_j^n - X_i^n|^2 = l_e^2, \quad (7)$$

where $l_e \neq l_o$. Equation 6 may be written in expanded form using the identities (3) to (7),

$$l_o^2 = \left| (X_j^n - X_i^n) + (V_j^{n-\frac{1}{2}} - V_i^{n-\frac{1}{2}}) \delta t + \left(\frac{F_j^n}{m_j} - \frac{F_i^n}{m_i} \right) (\delta t)^2 - \frac{M_{ij} \delta F_{ij}^n (\delta t)^2}{m_i m_j} \right|^2, \quad (8)$$

where $M_{ij} = m_i + m_j$ and l_e is incorporated in δF_{ij}^n . Because the constraint force acts along the interparticle bond, δF_{ij}^n has the form

$$\delta F_{ij}^n = \frac{m_i m_j}{M_{ij}} \left(\frac{l_e}{l_e} \right) a, \quad (9)$$

where a is a function of the velocities of particles i and j , the forces acting upon them, the actual separation l_e of the particles, and the specified constrained distance l_o . Equation 8 can be rewritten as

$$l_o^2 = \left| l_e + \Delta l - \frac{(\delta t)^2 a l_e}{l_e} \right|^2, \quad (10)$$

where

$$\Delta l = (\mathbf{V}_j^{n-\frac{1}{2}} - \mathbf{V}_i^{n-\frac{1}{2}}) \delta t + \left(\frac{\mathbf{F}_j^n}{m_j} - \frac{\mathbf{F}_i^n}{m_i} \right) (\delta t)^2. \quad (11)$$

Letting

$$b = \frac{a(\delta t)^2}{l_e}, \quad (12)$$

Eq. (10) can be rewritten as a quadratic equation in b ,

$$b^2 - 2 \left(1 + \frac{l_e \cdot \Delta l}{l_e^2} \right) b + \left(1 + \frac{|\Delta l|^2 + 2 l_e \cdot \Delta l - l_o^2}{l_e^2} \right) = 0. \quad (13)$$

Solving for b , an expression for a follows from Eq. (12):

$$a = \frac{1}{(\delta t)^2} \left[l_e + \frac{l_e \cdot \Delta l}{l_e} \pm \sqrt{l_o^2 + \frac{(l_e \cdot \Delta l)^2}{l_e^2} - (\Delta l)^2} \right]. \quad (14)$$

Substituting a into Eq. (9) (taking the negative root in Eq. (14)), we obtain the constraint force function

$$\delta F_{ij} = \frac{m_i m_j l_e}{M_{ij} (\delta t)^2} \left[1 + \frac{l_e \cdot \Delta l}{l_e^2} - \sqrt{\frac{l_o^2}{l_e^2} + \frac{(l_e \cdot \Delta l)^2}{l_e^4} - \frac{(\Delta l)^2}{l_e^2}} \right]. \quad (15)$$

A geometric interpretation of the constraint force function Eq. (15) is given in Appendix 2. The negative root in Eq. (15) gives the physically reasonable value for a . The square-root term in Eq. (15) is the magnitude of the projection of the displacement vector at time $t + \delta t$ onto the displacement vector at time t . Because $\Delta l \ll l_e$, the constraint force function is given by the approximation

$$\delta F_{ij} = \frac{m_i m_j l_e}{M_{ij}(\delta t)^2} \left[\frac{1}{2} + \frac{l_e \cdot \Delta l}{l_e^2} - \frac{1}{2} \frac{l_0^2}{l_e^2} \right], \quad (16)$$

which for a given timestep is less accurate than Eq. (15), but which is more computationally efficient because it is not necessary to compute a square root.

The constraint force function δF_{ij} given by Eq. (15) is sufficient for maintaining the specified bond distance l_0 if δF_{ij} is applied to a system of two rigidly bound particles. For a system of particles where each particle may be bound to more than one other particle, δF_{ij} given by Eq. (15) is not sufficient for maintaining bond distances because it assumes only a two-body constraint condition and neglects the cumulative effect of multiple connections. The constraint force function Eq. (15) is, however, the basis of an iterative procedure that maintains multiple rigid connections and also counteracts the accumulation of errors due to discretization or approximations in the constraint force functions. For a small number of iterations, the computed constraint force function may not converge completely to its limiting value every timestep.

The resulting small error in the computed value of δF_{ij} combined with errors resulting from discretization produce two types of constraint decay: deviations of the bond lengths from their assigned values and nonzero velocity between constrained pairs of particles along the direction of the bond. The constraint force function Eq. (15) has two features that counteract constraint decay. These are the conditions imposed by Eq. (6) that helps δF_{ij} to maintain an assigned bond length, and the contribution of the first term in Δl (given by Eq. (11)) to δF_{ij} that helps to minimize any relative

velocity components along bonds. These two factors contribute to the stability of the algorithm and give the algorithm the ability to correct for the accumulation of small errors resulting from discretization or any approximations in δF_{ij} .

The second term in Δl , given by Eq. (11), provides the feedback mechanism for a convergent iterative procedure for multiple connections. Once a constraint force for a particular bond is calculated, it is treated as an external force acting on the pair of particles. Thus at successive iterations, all the constraint forces maintaining other bonds attached to either of the two linked particles add to the constraint force maintaining that bond.

The mathematical basis of the algorithm presented here is the same as that for all other iterative constraint algorithms. For each multiply connected particle in the system, a force ΔF_i , the sum of all constraint forces acting on the particle i , is added to the total force F_i . The quantity ΔF_i may be given by

$$\Delta F_i = \sum_{j=1}^{K_i} \alpha_{ij} l_{ij}, \quad (17)$$

where the α_{ij} are constants that specify the magnitude of each constraint force and the l_{ij} are vectors that specify the K_i bonds linked to each particle i . If there are N bonds, the constrained system has N unknowns α_{ij} . Because there are then N constraint conditions, the system is solvable. Memon et al.¹³ have discussed efficient matrix methods for determining these unknowns using constraint equations obtained using the leapfrog integration scheme. Ryckaert et al.⁶ have pointed out that because the solution to the constraint system of equations is unique, any convergent procedure that satisfies all geometric constraint conditions by displacements of the form

$$\Delta X_i = \sum_{j=1}^{K_i} \frac{\alpha_{ij}(\delta t)^2}{m_i} l_{ij}, \quad (18)$$

is equivalent to results obtained through direct solution, i.e., the matrix method. It-

erative constraint algorithms differ in that some adjust Eq. (17), e.g., Memon et al.¹³, and others adjust Eq. (18), e.g., SHAKE^{6,9}. Each algorithm, however, permits different types of programming structures which may be optimal for particular types of problems and computers. We find that by using an exact two-particle algorithm as the basis of the MCF iteration, the accuracy and convergence are both improved, and convenient and efficient programming structures can be used to evaluate constraint forces.

An important aspect of the MCF algorithm is that it retains the reversibility property of the leapfrog algorithm. This is an important guarantee of qualitatively and quantitatively faithful Hamiltonian behavior^{3,4}, as previously shown for integration of relativistic charged particle orbits in electromagnetic fields.

Another important aspect of the MCF algorithm is that

$$\delta F_{ij}^{(i)} \longrightarrow 0 \quad (19)$$

for increasing (i) , independent of the order in which the δF_{ij} are computed. The superscript (i) in Eq. (19) designates the value of the constraint force function at the i th iteration. This property provides flexibility for computing δF_{ij} at each iteration because it is possible to partition the calculation of constraint force functions according to what is more efficient for a given problem or computer. Further, $F_{ij}^{(1)}$ can be evaluated unambiguously before the iteration begins.

The constraint force function F_{ij} is an explicit function of all forces acting on the linked particles. An important optimization permitted by the MCF algorithm, as well as other iterative constraint procedures such as SHAKE^{6,9}, is to set the forces between all linked particles to zero. This step reduces the magnitude of the maximum gradient of the constraint forces evaluated for the system and larger timesteps can be used.

III. Procedure for Calculating Constraint Forces

A flow chart describing the MCF algorithm is displayed in Fig. 1. For each particle in the system:

- (1) Compute the sum of all the forces acting on each particle due to the other particles in the system.

One important feature of this stage of the calculation is that the force between constrained particles is set to zero. This is not necessary in principle because the constraint force counteracts any mutual interaction. However, short-range interactions, typically at bond-length separations, are relatively large and result in a large value for terms containing $l_e \cdot \Delta l_e$ in the constraint force function. Because the gradient of the force is large at short range, if we included these short-range interactions, we would need a very small timestep.

- (2) Compute the displacement vector Δl given by Eq. (11) for each bond in the system.

The computation of Δl for the various bonds must be independent of the order of computation in any convergent iterative scheme. That is, the convergence does not depend on whether Δl is computed for the different bonds according to a specific procedure. Although different modes of computing the displacement vector Δl might result in different sets of values of Δl for the first iteration, successive iterations should diminish this difference.

- (3) The constraint force for maintaining each constrained distance is computed according to the constraint force function Eq. (15) or Eq. (16).
- (4) The constraint force for each constrained distance is added, with the appropriate sign, to the total force acting on the two constrained particles.

- (5) If the maximum constraint force calculated in the i th iteration is less than the convergence condition, or if the iteration counter exceeds the iteration limit, proceed to integrate the equations of motion; otherwise, repeat steps (2) to (4).

The development of the constraint forces during the first two iterations is illustrated in Fig. 2. For this example, we have chosen to compute the displacement vector sequentially from top to bottom. As the timestep δt decreases, the order of evaluation has less of an effect on the total force eventually applied to each particle. The constraint conditions are coupled to one another by treating constraint forces as external forces for subsequent calculations. Thus, while $\delta F_{12}^{(1)}$ depends only on external forces, $F_{23}^{(1)}$ depends on external forces and the constraint force $\delta F_{12}^{(1)}$. On the second iteration, $F_{12}^{(2)}$ depends on the external forces and $F_{23}^{(1)}$. This coupling leads to overall convergence of the constraint force function.

IV. Tests of the Algorithm

We conducted a series of simulations to test our algorithm. The purpose of these simulations was to

- (1) examine the stability and accuracy of the algorithm;
- (2) test those aspects of the algorithm that contribute to its general efficiency;
- (3) generate an initial version of a computer program based on this algorithm to be used for large scale simulations.

The model consisted of a 12×12 array of rigid tetra-atomic molecules in three-dimensional space. The configuration of each tetra-atomic molecule is shown in Fig. 3. The bond lengths and angles were fixed at the normal carbon-carbon single bond length of 1.54 Å and at the tetrahedral angle, respectively. Non-bonded interactions were given by a Lennard-Jones potential with parameters taken from atomic nitrogen¹⁶, $\epsilon = 0.5143 \times 10^{-14}$ erg and $\sigma = 3.310$ Å. The boundary conditions were periodic in the plane of the initial array. The system was confined in the third dimension by a pair of walls with the same LJ parameters as the particles themselves. Each molecule in the system was given a random initial velocity, and the motion was calculated with timesteps varying between 2×10^{-16} and 2×10^{-14} s. The deviation of actual distances (l_e) from their proper values (l_o) was monitored during each calculation. The number of distances deviating by more than 1% and the identity of the deviating pairs of particles were stored.

Figure 4 shows the system in its initial configuration, i.e., $t = 0$. Figures 4a, 4b and 4c are the XZ, YZ, and XY projections of the system, respectively, at $t = 0$. The "periodic boundaries" are at 0 Å and 60 Å along X and Y, while the walls in Z are at -4 Å and +8 Å.

The energy and constraint deviations are displayed in Fig. 5a and Fig. 5b, respectively, for a calculation of 42000 steps of 5×10^{-16} s each, using the complete constraint force function Eq. (15). The constraint force was evaluated with five iterations at each

timestep. The mean velocity of the atoms and centers of mass of the molecules, along with the corresponding standard deviations, are given in Table 1 (state 1). Also shown are the values of the mean and standard deviation of the total, kinetic, and potential energies. The reader will note the stability and rather narrow range of fluctuations in all these values. Figure 6b shows the number of constrained bonds that deviate from their proper values by more than 1%. The maximum deviation found under these conditions was about 2%. The average number of deviations per timestep is about 3, and the maximum recorded is 12, out of 720 bonds in the whole system. The identity of the constrained particles responsible for these deviations does not stay the same for many timesteps. Both the small number and the fluctuating identity of deviations demonstrate the ability of the MCF algorithm to counteract constraint decay. Under these conditions, we prefer to speak of "constraint fluctuation," since there is no tendency for the number or magnitude of the deviations to grow.

A similar calculation of 20000 steps using the approximate version of the constraint force function given by Eq. (16) is presented in Fig. 6. The state of the system (state 1, in Table 1) remained the same, well within the standard deviations. The number of constraint violations at the 1% level is a factor of thirty larger, but the magnitude of any single deviation is about the same, the largest deviation observed being about 2%. In this case too, the identity of the deviating bonds persisted over approximately 100 timesteps.

Figure 7 shows the energies of a system described by the parameters listed for state 2 in Table 1 over 3000 steps of 1×10^{-15} s (3×10^{-12} s total time) with the constraint force function evaluated over 10 iterations *in reverse order* from the other runs shown thus far. This shows that the stability of the algorithm does not depend on the order of the evaluation of the constraint force. A comparison of the calculation shown in Fig. 7 and a calculation starting from the same initial state, but with the constraint forces calculated in the standard sequence, is shown in Table 2. Although the centers of mass

of the molecules move 4.30 \AA on average, the final states corresponding to the "forward" and "backward" evaluation differ by 1 part in 10^5 in the mean displacements. The mean velocities and energies of the particles and molecules show a similar agreement. Table 2 also presents the results for forward versus backward runs at timesteps of $1 \times 10^{-14} \text{ s}$ and $2.5 \times 10^{-16} \text{ s}$. Although the total energy is conserved for all runs, the $1 \times 10^{-14} \text{ s}$ timestep is too large under these circumstances and there is a significant dependence on the order of evaluating the constraint force functions.

Figure 8 shows the absolute magnitude of the constraint force as a function of iteration for one timestep of $2 \times 10^{-16} \text{ sec}$, in state 2 of Table 1. The solid lines show the values of the largest constraint forces in the entire system, and the dashed line shows the average constraint force. The average constraint force is approximately an order of magnitude less than the maximum constraint force. Also, both the average and the maximum fall off exponentially at about the same rate of two orders of magnitude in ten iterations. Furthermore, the standard deviation about the mean constraint force is of the same order as the mean force itself (Table 3). Thus $\max(\delta F_{ij}^{(i)})$ appears to be a good diagnostic for the convergence of the constraint force evaluation. Figure 8 also shows that the constraint force does not decrease monotonically with iteration number, and that the identity of the pair of particles requiring the maximum constraint force changes with the number of iterations. An examination of the behavior of the constraint force function over many timesteps shows that those particle pairs requiring the largest constraint forces at early iterations change identity slowly, over several hundred timesteps under the present conditions (state 2), and that the magnitude of these forces also changes slowly. These last observations suggest that a predictor-corrector version of the algorithm, in which a previous evaluation of the constraint force is used as the initial estimate, would be very efficient. Preliminary tests support this idea.

V. Conclusion

We have derived and tested a new algorithm for constrained dynamics. The numerical basis of the algorithm is established by applying a constraint condition to the equations of motion in the central-difference leapfrog integration scheme, and solving for the forces of constraint. A geometric interpretation is provided in Appendix 2 that provides physical insight to the terms of the constraint force function. An approximate version of the constraint force function that eliminates the need to evaluate a square root saves approximately 10% in execution time. However, the ultimate cost of the decrease in accuracy has yet to be assessed. The algorithm, which is exact for two-body systems such as rigid rotors, is applied iteratively in simulations of polyatomic molecules. At the end of each iteration, the calculated constraint forces are added to the forces acting on the system. The largest constraint force falls off exponentially with iteration number, and may be used as a diagnostic of convergence.

We are developing and testing several simple extensions and improvements of the MCF algorithm that seem worthy of consideration for future implementation. Storing the composite constants $\{\alpha_{ij}\}$ for each of the constrained distances, rather than the total constraint force derived from these constants, allows calculation of better approximations to the constraint force in the first iteration at the next timestep, when the geometry has changed somewhat. Using the total constraint force suffers from the fact that the orientation of the atoms can be different and hence the vector constraint force must change, even though its magnitude along the line of centers may be identical. Using the constants $\{\alpha_{ij}\}$ has the added advantage that only one scalar constant has to be stored for each constrained distance rather than the three components of the corresponding vector force.

If the constraint constants $\{\alpha_{ij}\}$ are kept for the two previous steps rather than one, a simple extrapolation to the next timestep should give a good (linear) approximation to the inevitable changes in the values of $\{\alpha_{ij}\}$ that occur. This extrapolation should

work best for the slowly changing parts of $\{\alpha_{ij}\}$ for which the most iterations are required. The iteration procedure treats the rapid changes in the constraint constants that arise from close interactions and impulses with inherent efficiency. Thus it should be possible to reduce the number of iterations required appreciably, with improved accuracy, at essentially no additional computer cost.

The approximation to the square root used in Eq. (16) reduces the overall computation noticeably; but, as the differences between figures 5 and 6 imply, it is rather crude. Since a few iterations are necessary to obtain the constraint force anyway, the solution can be obtained to the roundoff limit by using a quadratically convergent iterative approximation to the square root. This procedure should be more efficient than Eq. (15) and equally accurate.

The constrained equations of motion obtained here are ultimately the same as those obtained by other constraint algorithms, an especially clear summary of which is presented by Levitt and Meirovitch¹⁷. A constraint force evaluated at a previous timestep can be stored as a particle attribute to be used for subsequent information processing, as in predictor-corrector schemes¹⁸ or interparticle state transitions¹⁹.

Although much of our approach has been influenced by the work of Edberg et al.¹¹, particularly the concept of penalty functions, such functions are not necessary in our formulation. We attribute this to the remarkable stability, demonstrated in part IV of this paper, of the reversible leapfrog integration scheme. We note, however, that energy conservation is not a sufficient indicator of the accuracy of the algorithm. A better test is that the statistical properties sought should not be unduly perturbed by the choice of timestep size. We have also derived a simple relation between the maximum force gradient experienced by the system, and the maximum timestep to integrate the equations of motion accurately (see Appendix 1).

Finally, we have written a computer program that uses the MCF algorithm to conduct molecular dynamics simulations of large assemblies of molecules. Among the

issues we are addressing are: development of optimum programming structures based on the two-particle constraint force approach of the MCF algorithm; development of efficient vector procedures based on the property that the MCF method converges independent of the order in which the F_{ij} are computed; and extending the MCF method to predictor-corrector schemes for computing constraint forces as described above.

Acknowledgements

This work was supported by the Physics Division of the Office of Naval Research and by the Naval Research Laboratory. Dr. M. Nagumo was supported in part by ONR project N0001487WX24241. The authors would like to acknowledge the continued support of Dr. Robert Junker and the encouragements from Drs. Robert Wyatt and Herschel Rabitz.

References

1. L. Verlet, Phys. Rev., 159, 98 (1967).
2. D. Beeman, J. Comput. Phys. 20, 130 (1976).
3. J. Boris, Proceedings of the Fourth Conference on Numerical Simulation of Plasmas, p. 3 (1970).
4. O. Buneman, J. Comput. Phys., 1, 517 (1967).
5. R.W. Hockney and J.W. Eastwood, Computer Simulation Using Particles, McGraw-Hill Inc., New York (1981).
6. J.-P. Ryckaert, G. Ciccotti and H.J.C. Berendsen, J. Comput. Phys. 23, 327 (1977).
7. G. Ciccotti, M. Ferrario and J.-P. Ryckaert, Mol. Phys. 47, 1253 (1982).
8. J.-P. Ryckaert, Mol. Phys. 55, 549 (1985).
9. W.F. van Gunsteren and H.J.C. Berendsen, Mol. Phys. 34, 1311 (1977).
10. W.F. van Gunsteren, Mol. Phys. 40, 1015 (1982).
11. R. Edberg, D.J. Evans and C.P. Morris, J. Chem. Phys. 84, 6933 (1986).
12. D.J. Evans, W.G. Hoover, B.H. Failor, B. Moran and A.J.C. Ladd, Phys. Rev. A28, 1016 (1983).
13. M.K. Memon, R.W. Hockney and S.K. Mitra, J. Comput. Phys. 43, 345 (1981).
14. K. Singer, A.J. Taylor and J.V.L. Singer, Mol. Phys. 33, 1757 (1977).
15. C. Hoheisel, R. Vogelsang and M. Schoen, Comp. Phys. Comm., 43, 217 (1987).
16. P.S.Y. Cheung and J.G. Powles, Mol. Phys., 30, 921 (1975).
17. M. Levitt and H. Meirovitch, J. Mol. Biol. 168, 617 (1983).

18. S. Lambrakos, M. Nagumo, E.S. Oran and J.P. Boris, "A Predictor-Corrector Multiple Constraint Force Algorithm," (in preparation for submission to J. Comput. Phys.)
19. S.G. Lambrakos, E.S. Oran, J.P. Boris and R.H. Guirguis, Proceedings of the 1987 Topical Conference: Shock Waves in Condensed Matter, Monterey, California, June 20, 1987.

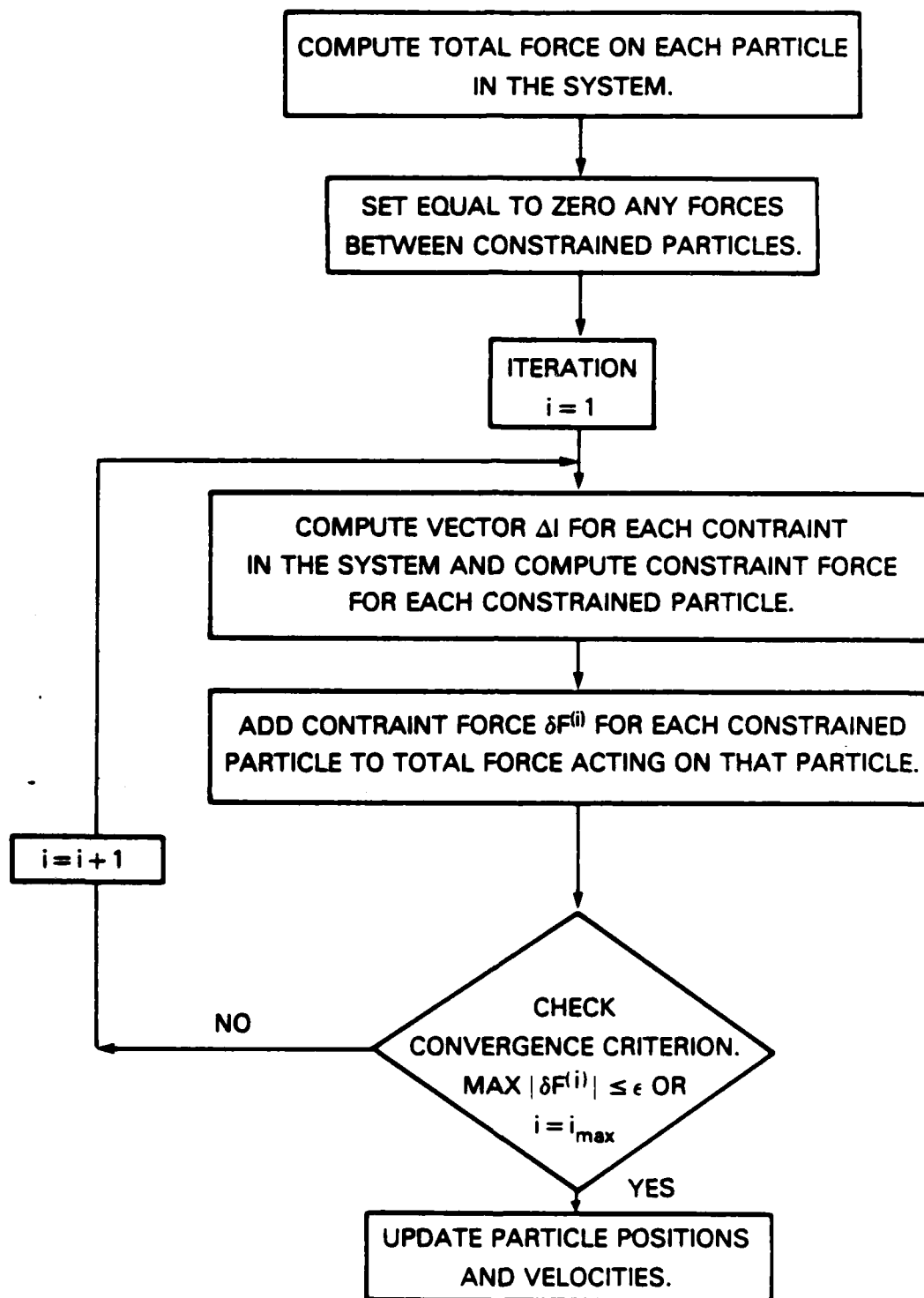


Figure 1. Sequence of operations required for maintaining constrained distances using Eqs. (11) and (15).

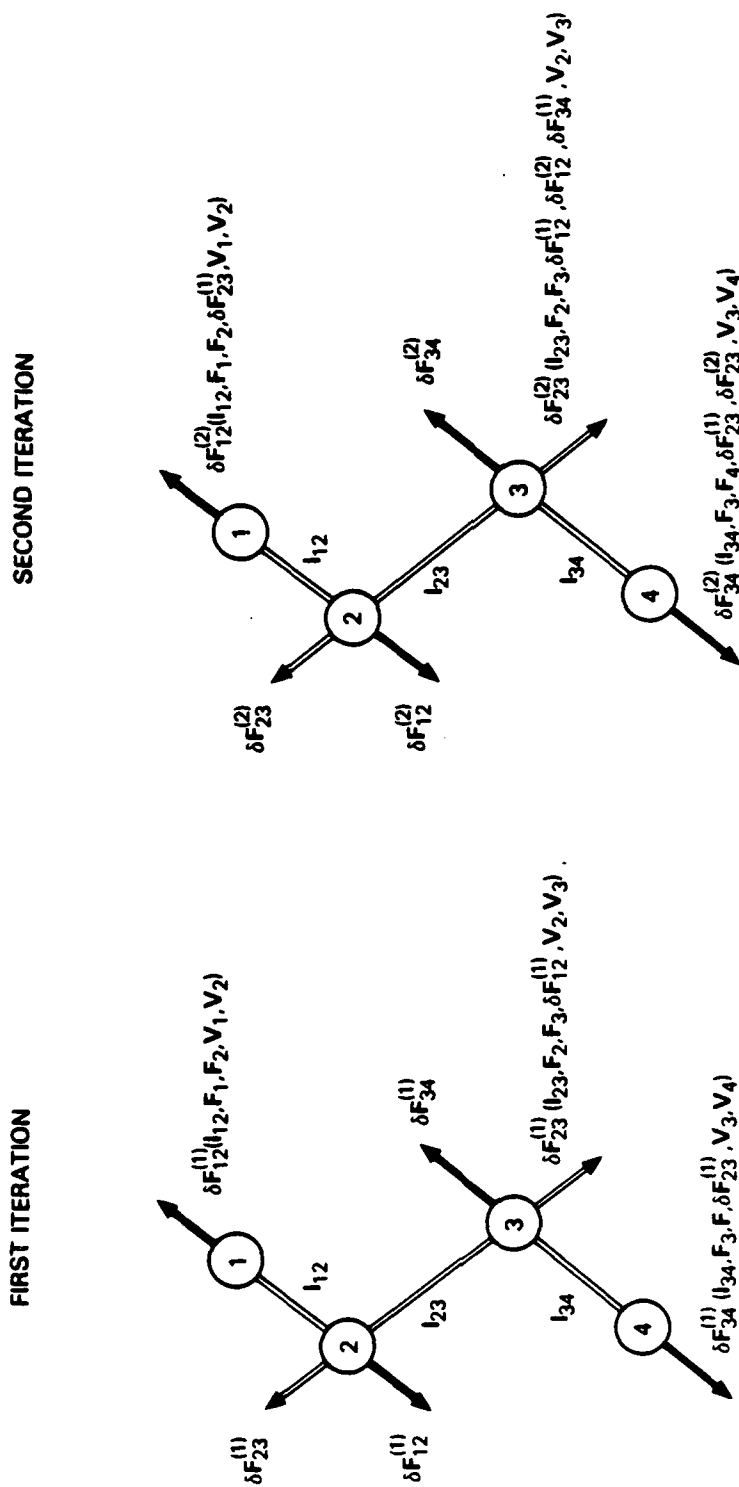


Figure 2. Constraint Forces contributing to the total force on each bound particle at different iterations. The quantity F_i is the total force on particle i not including constraint forces. The quantity $\delta F_{ij}^{(i)}$ is the constraint force for maintaining bond (ij) at the i th iteration.

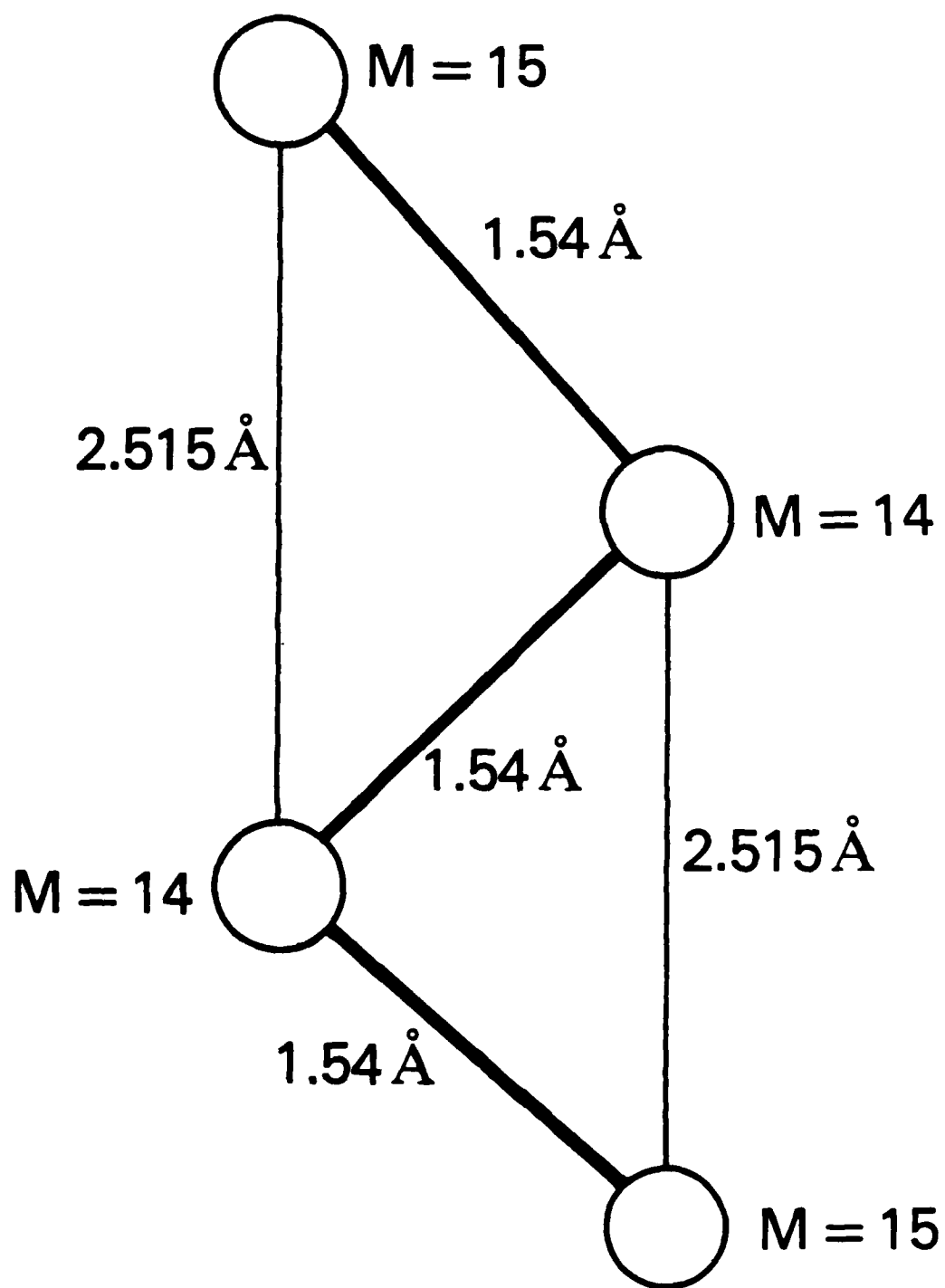


Figure 3. Configuration of bound system of particles to which constraint algorithm is applied.

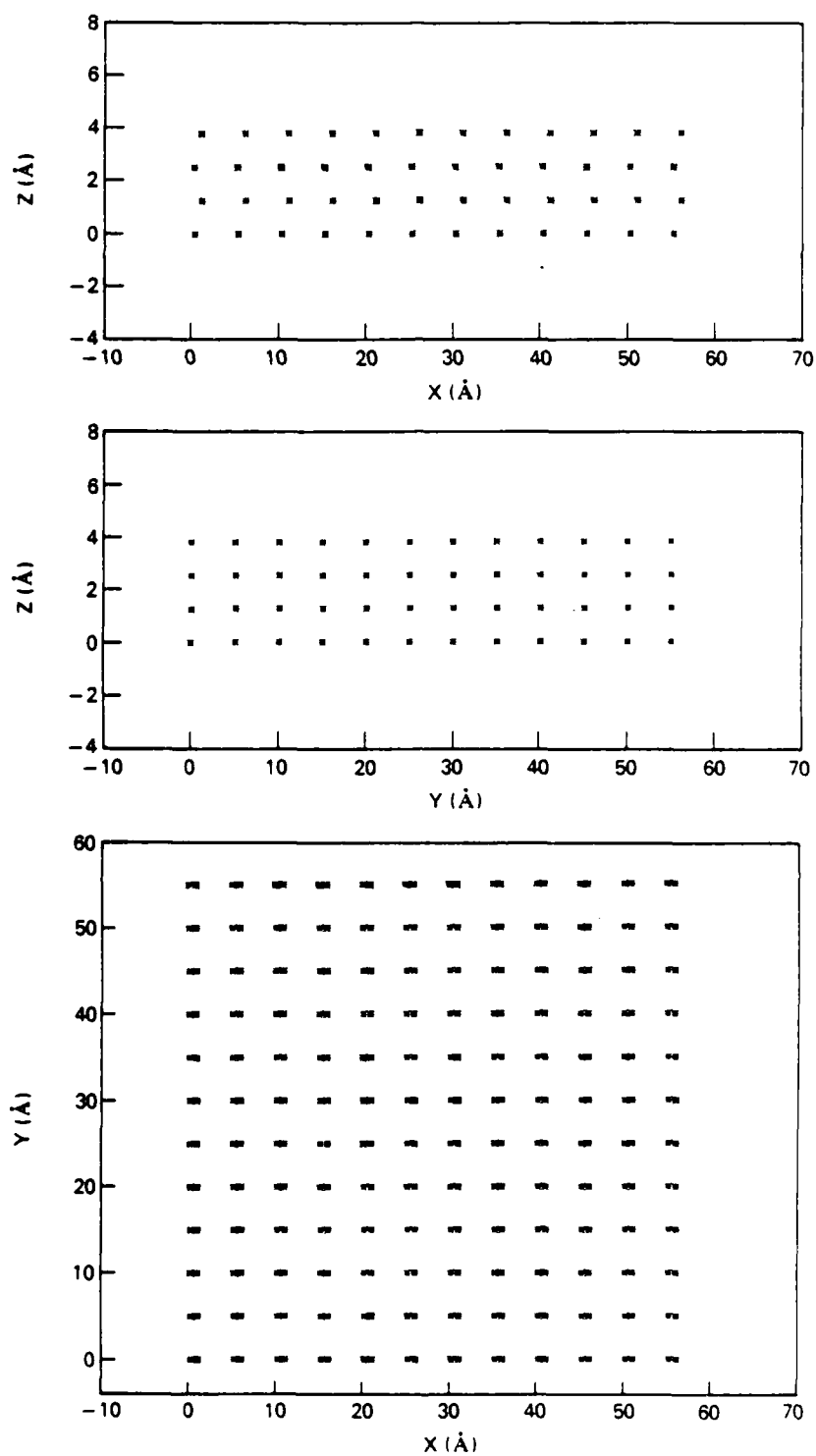


Figure 4. Projection of particle positions in XZ, YZ, and XY planes for initial configuration of system.

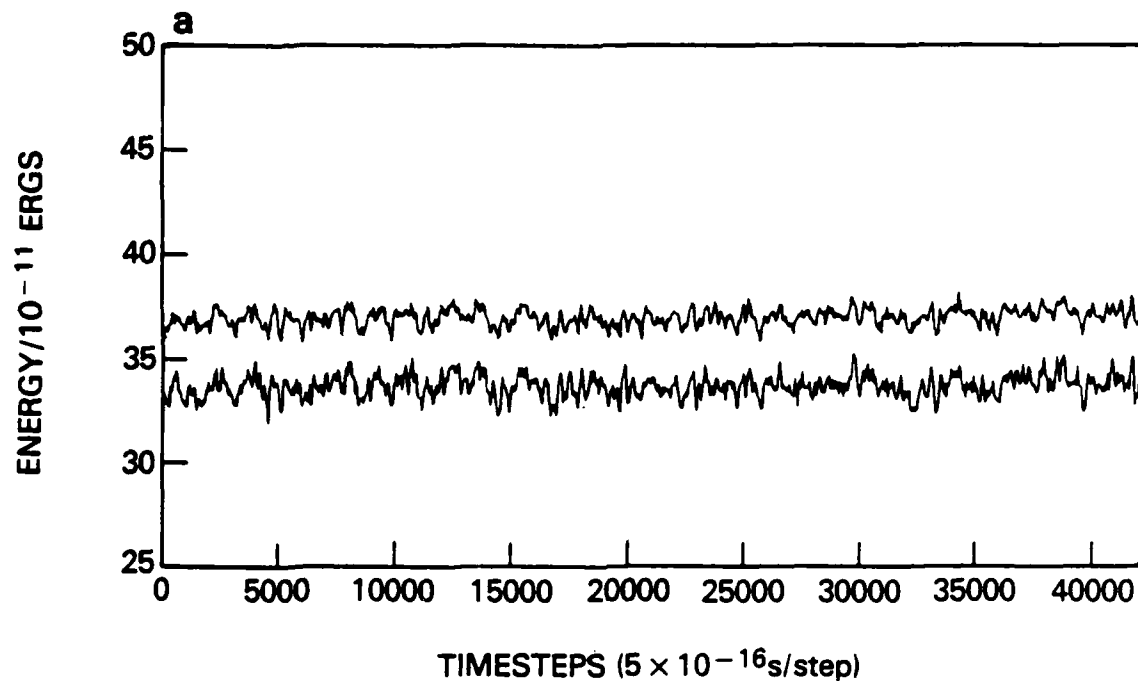


Figure 5a. Energy as a function of time. Upper curve - total energy (kinetic + potential). Lower curve - total kinetic energy. (The constraint force is given by exact expression Eq. (15)).

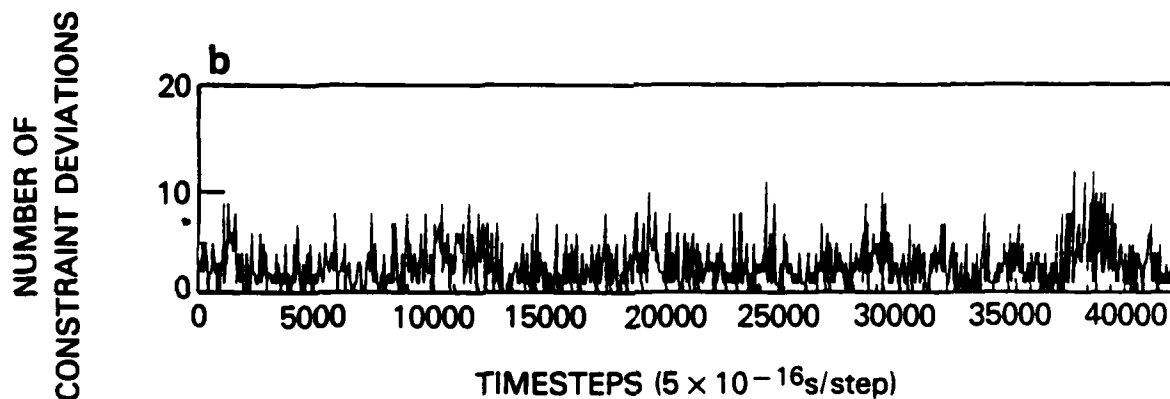


Figure 5b. Number of constraint deviations as a function of time. (The constraint force is given by exact expression Eq. (15)).

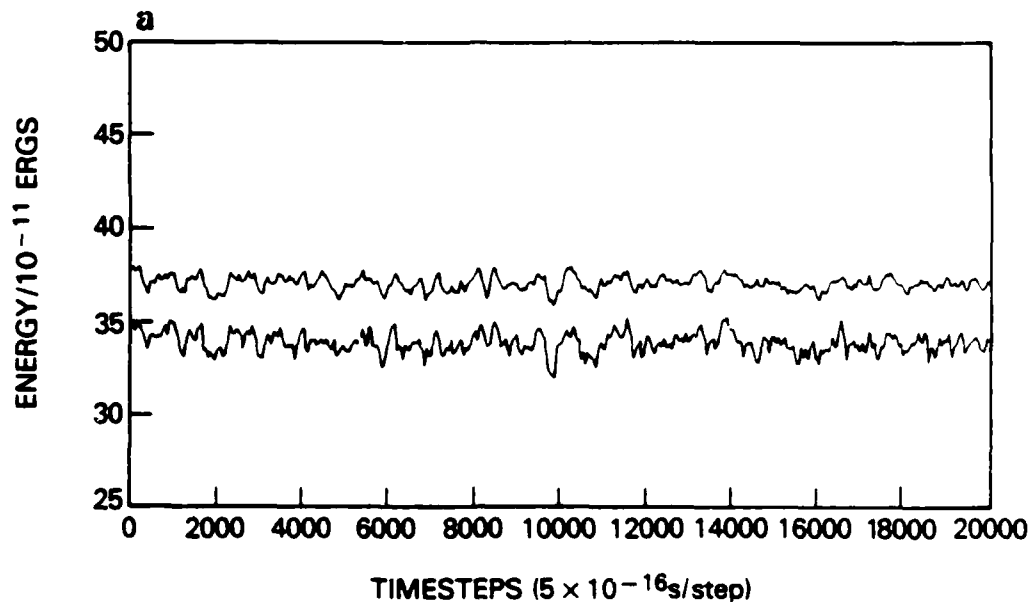


Figure 6a. Energy as a function of time. Upper curve - total energy (kinetic + potential). Lower curve - total kinetic energy. (The constraint force is given by approximate expression Eq. (16)).

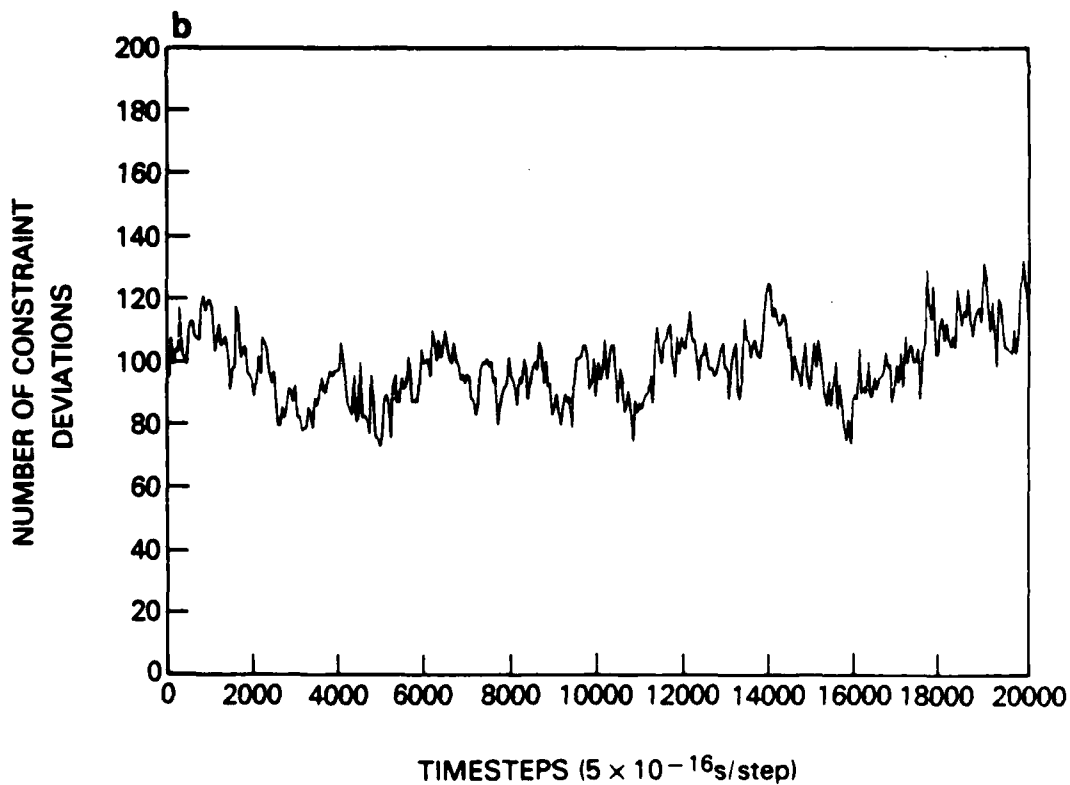


Figure 6b. Number of constraint deviations as a function of time. (The constraint force is given by approximate expression Eq. (16)).

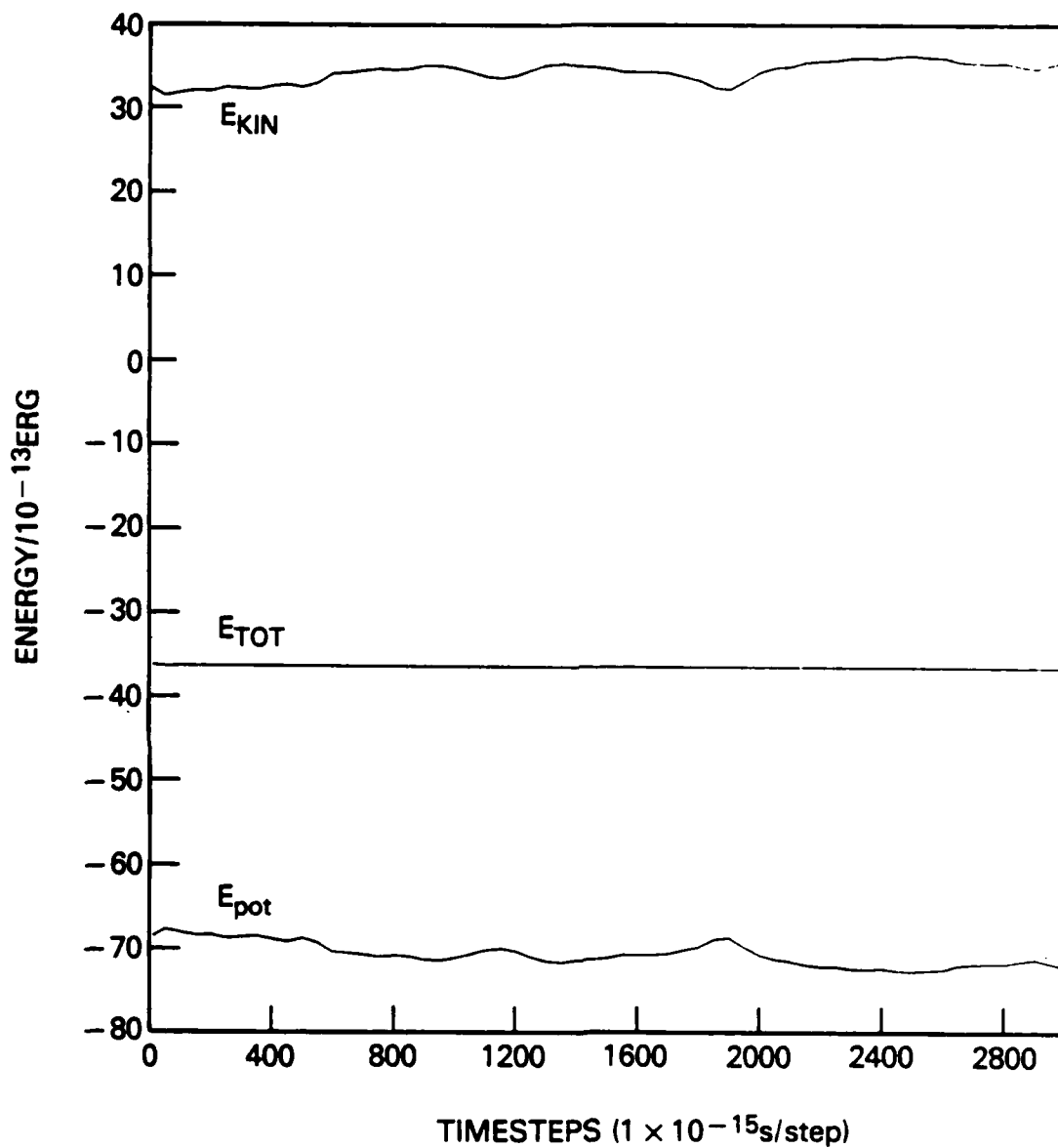


Figure 7. Total, kinetic, and potential energy of system as a function of time corresponding to a reversal of the order of evaluating the constraint forces. (Exact expression Eq. (15), 10 iterations).

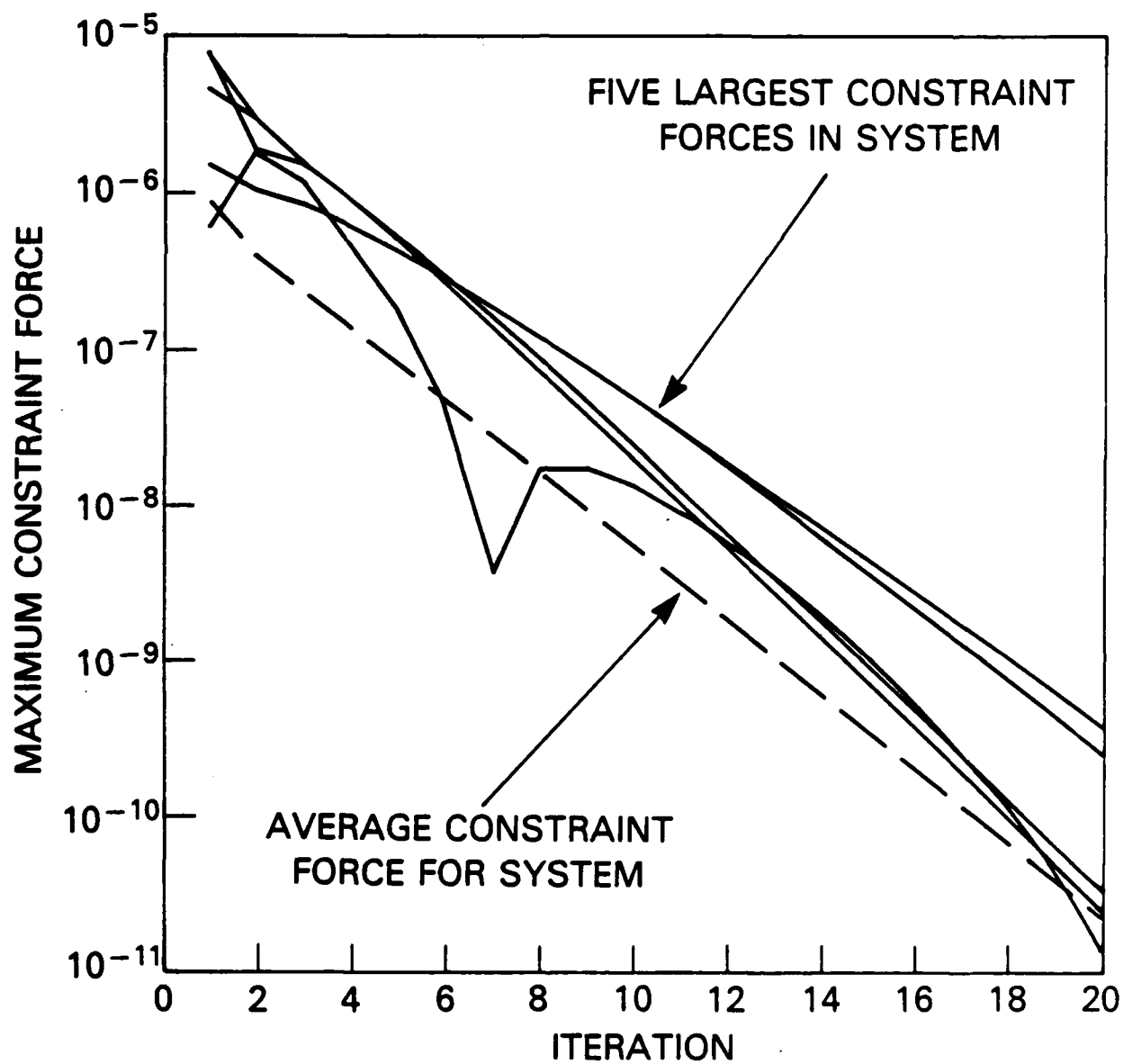


Figure 8. Maximum constraint force in system as a function of iteration.

Appendix 1.

Derivation of Eq. (2): Maximum Timestep for Accurately Integrating Equations of Motion

The trajectory of a particle of mass m in a force field $F(X)$ is determined by

$$m \frac{d^2 X}{dt^2} = F(X), \quad (A1-1)$$

where X is the position. In general, $F(X)$ is a nonlinear function of X . However, an accurate approximation of Eq. (A1-1) by difference equations depends on the form of $F(X)$ in some small neighborhood δX . Applying a perturbation to Eq. (A1-1) and expanding F in a Taylor series about X , we obtain

$$m \frac{d^2(X + \delta X)}{dt^2} = F(X + \delta X) \approx F(X) + \delta X \frac{dF}{dX}. \quad (A1-2)$$

Subtracting Eq. (A1-1) from Eq. (A1-2), we obtain

$$m \frac{d^2(\delta X)}{dt^2} \approx \delta X \frac{dF}{dX} \quad (A1-3)$$

The quantity $d^2(\delta X)/dt^2$ can be approximated by applying central differences to δX ,

$$m \frac{d^2(\delta X)}{dt^2} \approx m \frac{\delta X(t + \delta t) - 2\delta X(t) + \delta X(t - \delta t)}{\delta t^2}, \quad (A1-4)$$

for δt sufficiently small. Bounds on the size of δt are determined by expanding $\delta X(t \pm \delta t)$ in a Taylor series about t ,

$$\delta X(t \pm \delta t) \approx \delta X(t) \pm \delta t \frac{dX}{dt} + \frac{(\delta t)^2}{2!} \frac{d^2(\delta X)}{dt^2} \quad (A1-5)$$

$$\pm \frac{(\delta t)^3}{3!} \frac{d^3(\delta X)}{dt^3} + \frac{(\delta t)^4}{4!} \frac{d^4(\delta X)}{dt^4} \pm \dots$$

Substituting Eq. (A1-5) into the right side of Eq. (A1-4), we obtain

$$\frac{\delta X(t + \delta t) - 2\delta X(t) + \delta X(t - \delta t)}{\delta t^2} \approx \frac{d^2(\delta X)}{dt^2} + \frac{\delta t^2}{12} \frac{d^4(\delta X)}{dt^4}. \quad (A1-6)$$

Equation A1-6 shows that the approximation given by Eq. (A1-4) is accurate only if

$$\frac{\delta t^2}{12} \frac{d^4(\delta X)}{dt^4} \left(\frac{d^2(\delta X)}{dt^2} \right)^{-1} = \gamma, \quad (A1-7)$$

where $\gamma \ll 1$. It follows from Eq. (A1-3) that

$$\frac{d^4(\delta X)}{dt^4} \approx \frac{1}{m} \frac{d^2(\delta X)}{dt^2} \frac{dF}{dX}. \quad (A1-8)$$

Substituting Eq. (A1-8) into Eq. (A1-7) and letting $\alpha = (12m\gamma)^{1/2}$, we obtain

$$\delta t = \frac{\alpha}{\left[\frac{dF}{dX} \right]^{1/2}}, \quad (A1-9)$$

which is Eq. (2) in the text.

Appendix 2.

Geometric Interpretation of Eq. (15)

The constraint force maintaining a fixed bond length between two particles given by Eq. (15) can also be obtained by considering the relative motion of these particles. The motion of a particle of mass m_i relative to a particle of mass m_j is equivalent to the motion of a particle of mass $\mu = m_i m_j / (m_i + m_j)$ moving relative to a fixed point in space in the center-of-mass coordinate system. We therefore consider the force required to constrain the motion of a particle of mass μ to a fixed orbit about a point in space.

The constraint force function Eq. (15) may be expressed as the sum of two forces,

$$\delta F_{ij} = \delta F_{c1} + \delta F_{c2}. \quad (A2-1)$$

The force δF_{c1} counteracts the external forces that tend to cause deviations from the fixed bond length. The force δF_{c2} both gives the bound particle a trajectory consistent with the rigid-bond constraint and counteracts the influence of accumulated numerical errors that cause the bond length to deviate from its fixed value. The force δF_{c1} , given in terms of the geometric quantities defined in Section III,

$$\delta F_{c1} = \frac{\mu}{(\delta t)^2} \frac{l_e \cdot \Delta l}{l_e} \left(\frac{l_e}{l_e} \right), \quad (A2-2)$$

is explained by Fig. A1-1. The quantity Δl , given by Eq. (11), is the displacement vector of the particle in the center-of-mass system. As can be seen in Fig. A2-1, the quantity l_{NC} is the position of the particle at time $t + \delta t$ if the constraint force Eq. (A2-2) is not applied. The force δF_{c2} , where

$$\delta F_{c2} = \frac{\mu}{(\delta t)^2} (l_e - l_c), \quad (A2-3)$$

is also explained by Fig. A1-1. Note that in principle $|l_e| = |l_o|$, but in actual simulations the accumulation of numerical errors destroys this equality. The quantity l_e is the projection of the displacement vector at time $t + \delta t$ onto the displacement vector at time t . Because we desire a fixed-orbit trajectory of orbital radius $|l_o|$, the force Eq. (A2-3) must be applied with

$$l_e = \left(\frac{l_e}{l_e} \right) \sqrt{l_o^2 + \frac{(l_e \cdot \Delta l)^2}{l_e^2} - (\Delta l)^2}. \quad (A2 - 4)$$

Figure A1-1 gives a geometric description of Eq. (A2-4). Note that in addition to adjusting the trajectory in accordance with the constraint distance, the constraint force also adjusts the trajectory to correct for any deviations due to numerical error. Substituting Eq. (A2-4) into Eq. (A2-3) and adding Eq. (A2-2), we obtain Eq. (15).



Table 1. Average states of system used to test stability and accuracy of algorithm. State 1 refers to figures 6 and 7. State 2 refers to figures 8 and 9. Velocities are averaged over the entire system at one timestep. Energies are averaged over an entire run. State 1 energies averages are over 20000 timesteps; state 2 energy averages are over 3000 timesteps.

	Mean Particle Velocity (cm/s)	Mean Molecular Velocity (cm/s)	Total Energy (erg)	Kinetic Energy (erg)	Potential Energy (erg)
State 1:	1.98×10^5 $\pm 8.88 \times 10^4$	1.27×10^5 $\pm 4.91 \times 10^4$	3.66×10^{-10}	3.32×10^{-10}	3.38×10^{-10}
State 2:	2.05×10^4 $\pm 9.41 \times 10^3$	1.37×10^4 $\pm 5.52 \times 10^3$	-3.65×10^{-12}	3.41×10^{-12}	-7.05×10^{-12}

Table 2. Average displacements of particles, $\langle \delta r_{at} \rangle$, and centers-of-mass, $\langle \delta r_{cm} \rangle$, between forward (F) and reverse (R) evaluation of the constraint force functions. The total period simulated was 3×10^{-12} sec in all cases.

timestep direction	1×10^{-14} s		1×10^{-15} s		1×10^{-16} s	
	F	R	F	R	F	R
$\langle \delta r_{at} \rangle$	7.60 ± 14.92	7.45 ± 14.96	5.38 ± 12.96	5.38 ± 12.96	5.98 ± 14.07	5.98 ± 14.07
$\langle \delta r_{cm} \rangle$	5.77 ± 7.49	5.67 ± 7.47	4.31 ± 6.52	4.31 ± 6.52	4.44 ± 6.71	4.44 ± 6.71

Table 3. Maximum and average constraint force as a function of iteration.

Iteration	Bond Index	Maximum δF_{ij}	Average δF_{ij}
1	269	7.049×10^{-6}	9.735×10^{-7}
2	212	2.674×10^{-6}	4.387×10^{-7}
3	267	1.622×10^{-6}	2.690×10^{-7}
4	149	1.044×10^{-6}	1.605×10^{-7}
5	149	6.896×10^{-7}	9.524×10^{-8}
6	149	4.489×10^{-7}	5.692×10^{-8}
7	149	2.879×10^{-7}	3.397×10^{-8}
8	149	1.825×10^{-7}	2.028×10^{-8}
9	149	1.145×10^{-7}	1.205×10^{-8}
10	149	7.126×10^{-8}	7.120×10^{-9}
11	149	4.400×10^{-8}	4.201×10^{-9}
12	149	2.699×10^{-8}	2.473×10^{-9}
13	149	1.645×10^{-8}	1.454×10^{-9}
14	241	1.031×10^{-8}	8.555×10^{-10}
15	241	6.799×10^{-9}	5.030×10^{-10}
16	241	4.486×10^{-9}	2.964×10^{-10}
17	241	2.960×10^{-9}	1.753×10^{-10}
18	241	1.953×10^{-9}	1.040×10^{-10}
19	241	1.289×10^{-9}	6.175×10^{-11}
20	241	8.503×10^{-10}	3.678×10^{-11}

END

DATE

FILMED

5-88

DTIC